SDA- Shot Data Acquisition and Analysis

Example of major applications that use the Java DAE framework

A success story and a cautionary tale….

Jean Slaughter

Java Migration Review

Feb 4, 2004

What is SDA?

SDA is a system for acquiring, archiving and analyzing data from stores.

- Correlation of information from multiple sources at specific times during the stores
  - Information for day to day monitoring of stores
  - Specialized studies
- Two Aspects
  - Acquiring and archiving the data
  - Using the data
- Controls, Computing Division, coordinated in the Integration Department
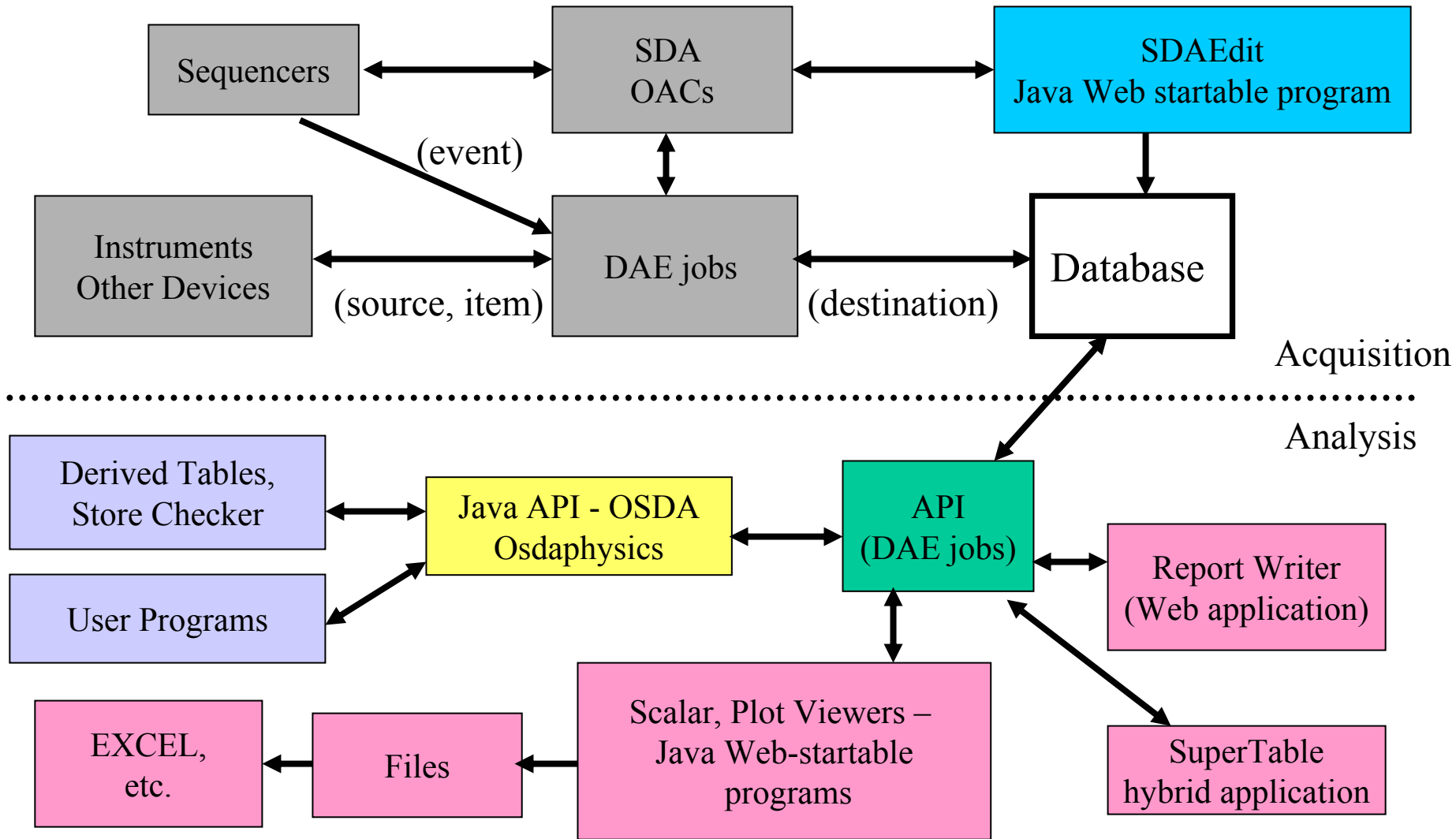- Java DAE framework

D44 versus SDA

- **Lumberjack data**
  - Fixed frequencies, i.e. non-triggered
  - Used for studies
  - Circular buffers – data is overwritten after 1-30 days
  - Any user can modify any archive configuration

- **SDA data**
  - Triggered at specific events
  - Only during stores*
  - Permanent archive
  - Standard content

Both types are needed to do SDA analysis

*store as opposed to studies. There are recycler only and pbar only "stores".

SDA Diagram

Sequencers ⟷ SDA OACs ⟷ SDAEdit Java Web startable program

Sequencers → (event) → DAE jobs

SDA OACs ⟷ DAE jobs

SDAEdit Java Web startable program → Database

Instruments Other Devices ⟷ DAE jobs

(source, item) DAE jobs (destination) Database

Acquisition

....

Analysis

Derived Tables, Store Checker ⟷ Java API - OSDA Osdaphysics ⟷ API (DAE jobs) ⟷ Report Writer (Web application)

User Programs ⟷ Java API - OSDA Osdaphysics

API (DAE jobs) ⟷ SuperTable hybrid application

API (DAE jobs) ⟷ Scalar, Plot Viewers – Java Web-startable programs

EXCEL, etc. ← Files ← Scalar, Plot Viewers – Java Web-startable programs

Comments on Data Acquisition in SDA - I

- Initially a long tail in DA times caused problems until system parameters were tuned.

- Data acquisition in certain cases/sets has been staggered.

- Because of this history, I worry about how close we are to system capacity.   We should add many more devices, like magnet settings, readbacks.

- While SDA is not "real time" in the sense of real time control, in many cases we do need to know the actual physical time of the event.

- Specs on system capacity and timing are at best vague. The users have not tried to stress it.

Comments on Data Acquisition in SDA - II

- Consolidation is a very important part of the current implementation - multiple simultaneous requests from many sources.
- Calibration database and SDA
  - SDA stores unscaled data because of data load.
  - Causes problems when the scale factors in the database change. Information from old stores gets the new scale factor..
  - Solution 1 – Have Brian Hendricks rescale the old data.
    - Not good
  - Solution 2 – For smart front ends
    - Change database scale factors to 1.0, offsets to 0.
    - Apply the scale change in the front end.
    - Supply the current scales, offsets as ACNET variables that are data logged and stored in SDA.
  - Vulnerable to unnoticed scale factor changes.

Overview of Analysis Applications

- Viewers – interactive browsers – Web startable
- Reports – flexible, GUI driven method of asking for a subset of data – Browser, JSP, applet
- OSDA API – Java classes to allow program access to data.
- Standard analysis jobs run every store
  - Luminosity plot
  - Store checker
  - Derived tables
  - SuperTable (ntuple?), Tev subset table
  - Derived emittances table
  - Efficiency table
- Variety of access implementations as we learned.
- Firewall influenced designs.

SDA Reports

SDA Reports (MT)

Report (MT) JSPs on tomcat

Reports description files

HTML Browser (IE, Netscape, Mozilla) +SVG Viewer (Adobe, Corel).

HTTP

DAE Jobs, callbacks

DAE (Data Acqusition Engine) **SQL & scaling**

Report Editor (Java applet)

Database (Sybase)
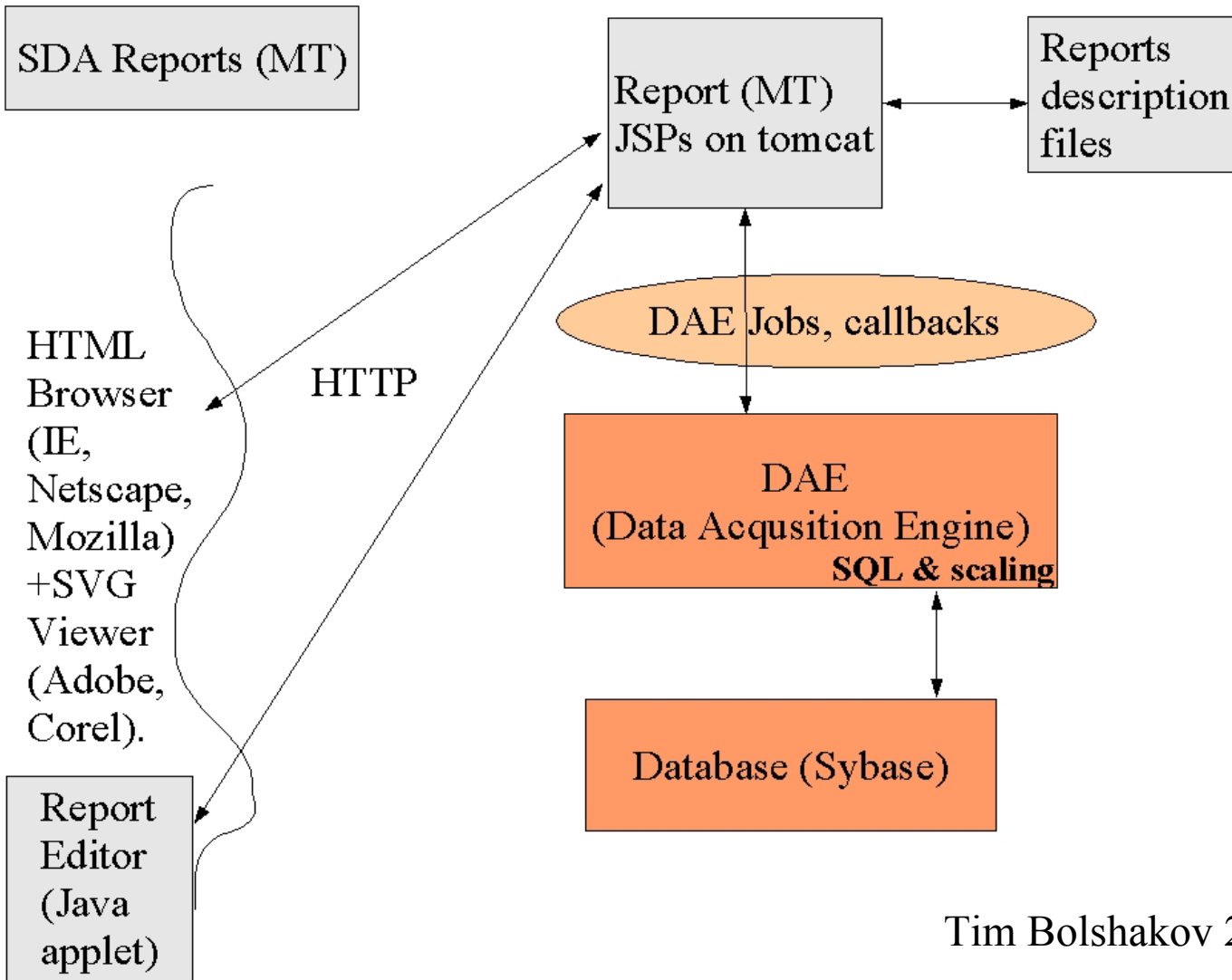
Tim Bolshakov 2/3/2004

OSDA (Offline SDA) and Osdaphysics

- **Original Conception of SDA**
  - Data Acquisition via Java DAEs
  - SDA Edit for specifying what devices to read and when.
  - Viewers, Reports make files for input to EXCEL, etc.
  - Data from front ends can be used as is.
- **Experience led to OSDA, Osdaphysics**
  - Data from front ends needs massaging. Want a single source for algorithms, including OACs.
  - Complex questions need direct program access to SDA data and D44 data.
- **Current status OSDA, Osdaphysics**
  - Evolved from special purpose analyses where flexibility was the prime consideration. Very slow.
  - Now used for operations - being upgraded for speed and maintainability (clarity).
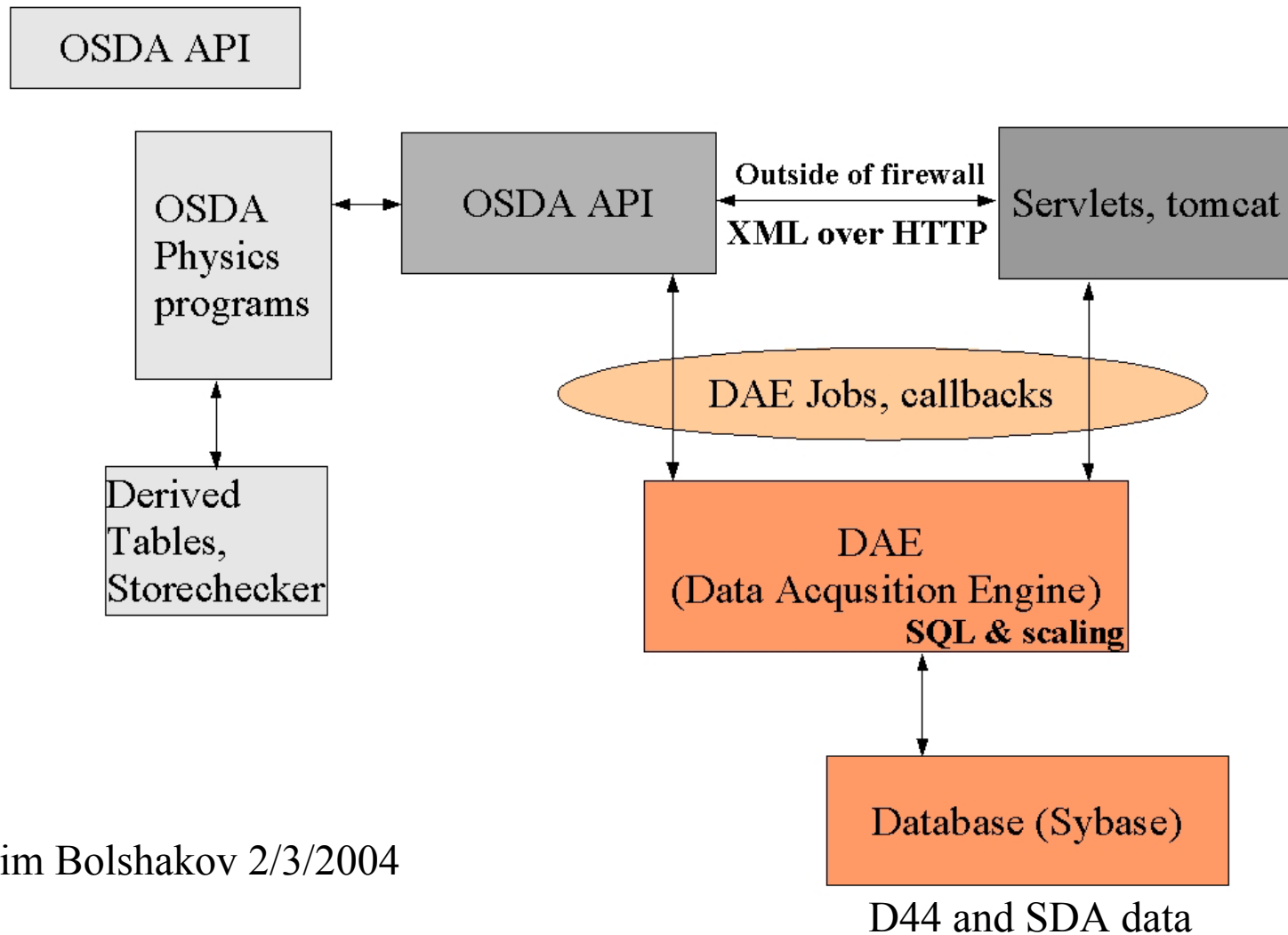
Tim Bolshakov 2/3/2004

Integrated Luminosity Table

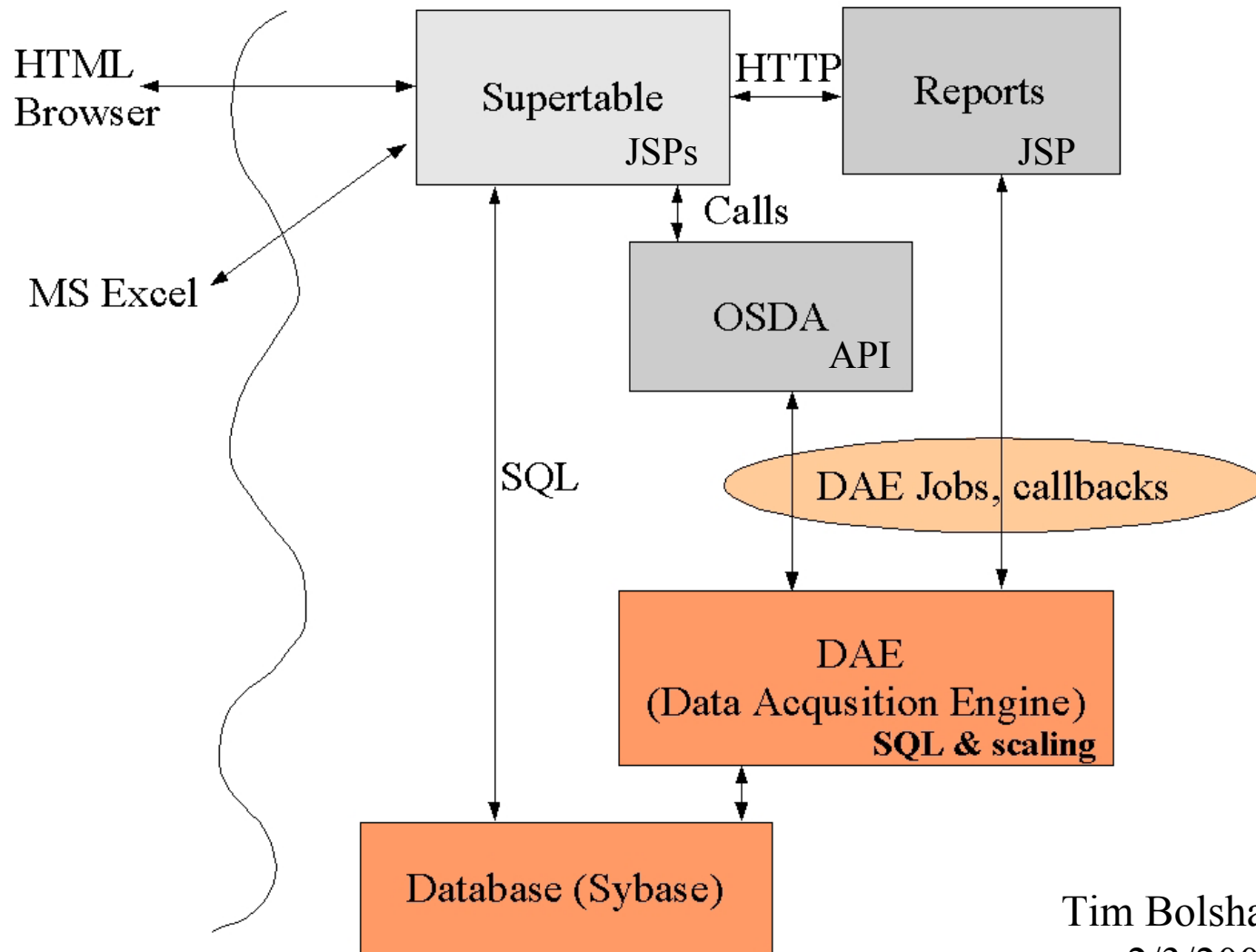SDA- Jean Slaughter – July 2003
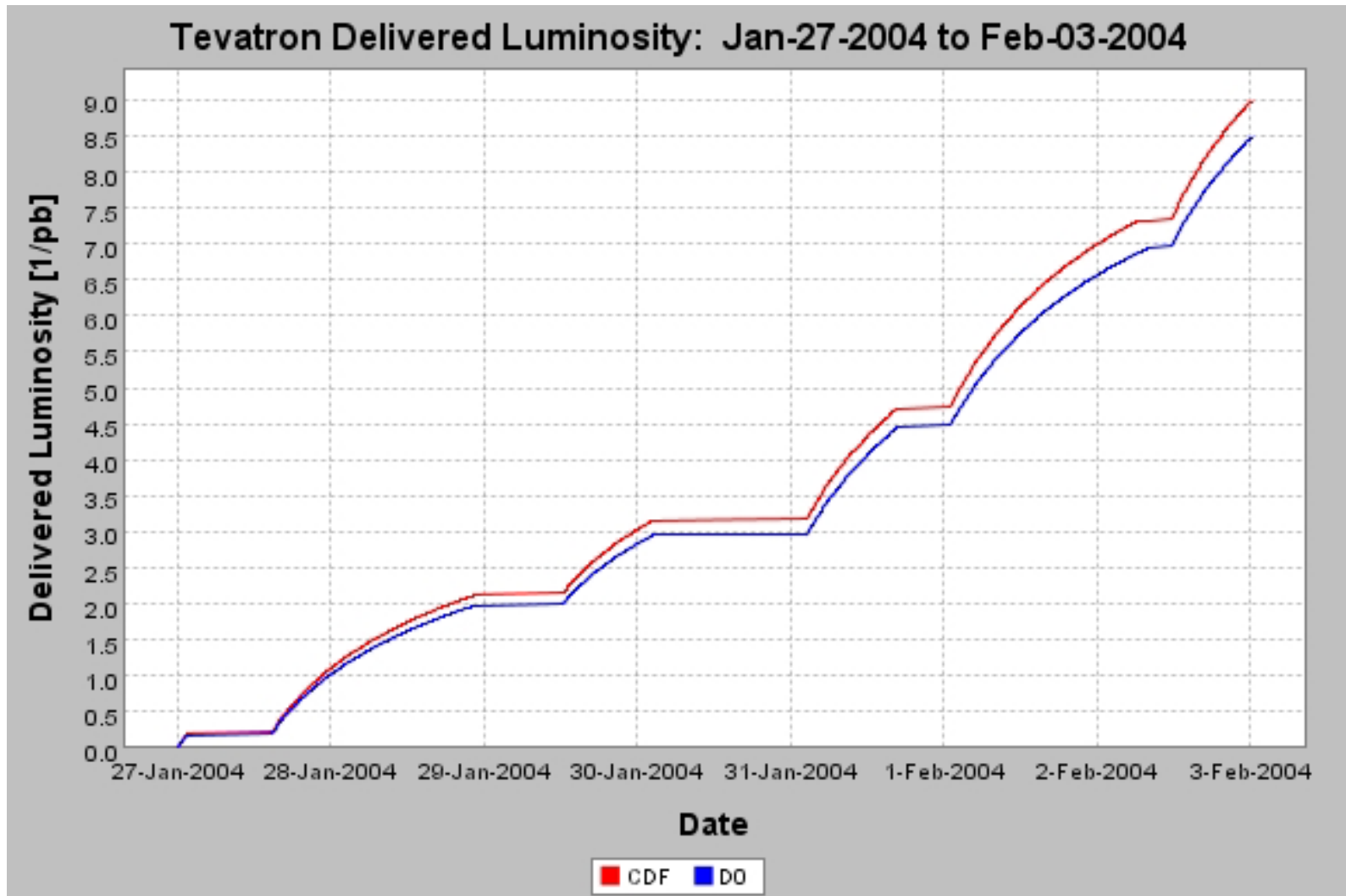
Derived Tables- Detailed Information on each Store

- Table per store – built automatically
  - All 6 emittances
  - Intensities
- Average and bunch by bunch information
- Use best algorithms to get physics quantities
  - Not always available directly from front-end.
- Uses OSDA, Osdaphysics
- Interactive plotting interface
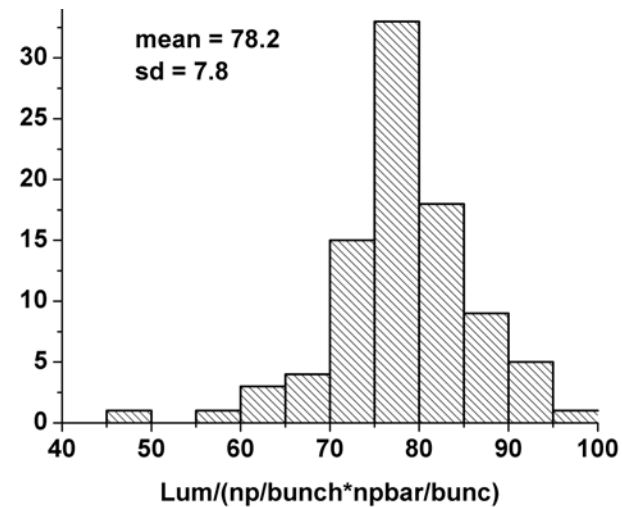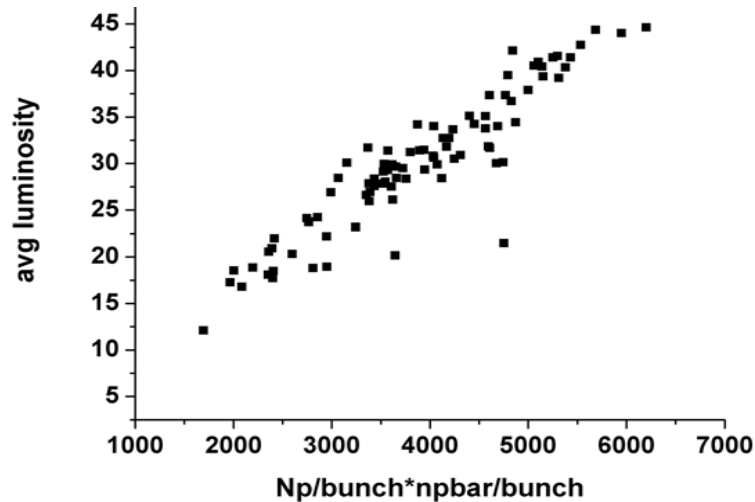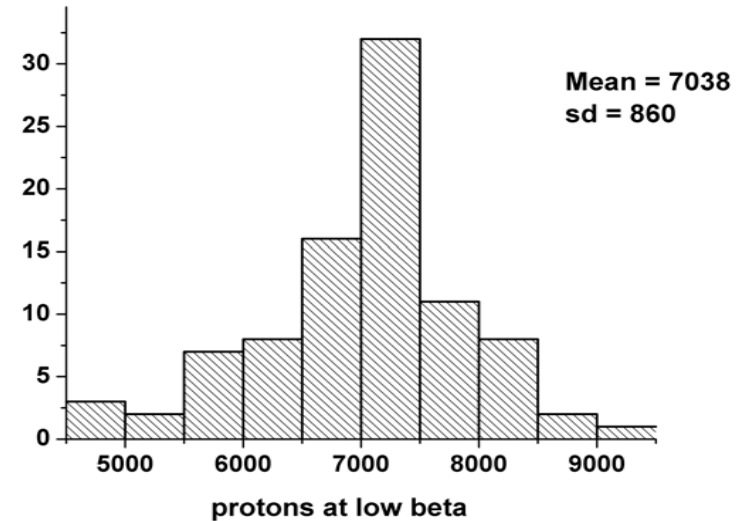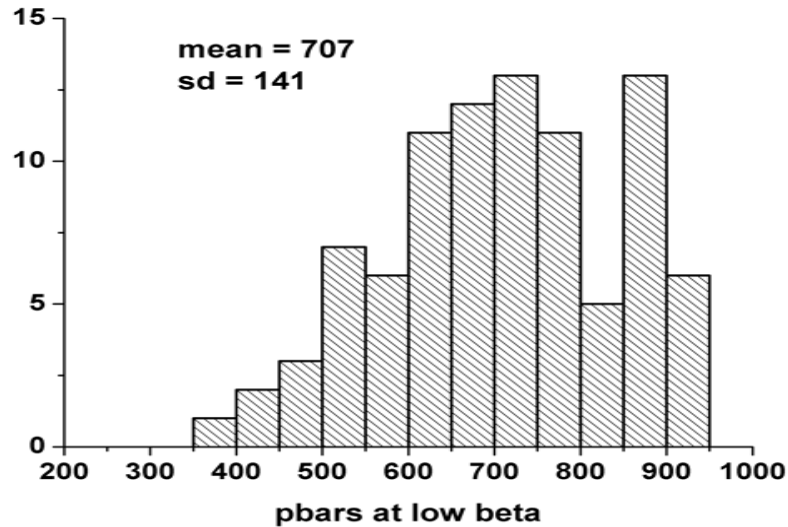- Not as useful as we expected.

Supertable

- One line per store
- Built automatically every store
- 130+ quantities of general interest
  - Dates, time on helix, length of store
  - How store ended
  - Luminosities, intensities, lifetimes
  - Efficiencies at each stage from p, pbar sources to HEP
  - Emittances at each stage
- Web Accessible – HTML, EXCEL
  - Used by all the bigwigs….
- Hybrid implementation
  - Complex computations
  - Rebuilt as understanding changes
  - Adapt to broken devices

SuperTable

HTML Browser

Supertable — JSPs

HTTP

Reports — JSP

MS Excel

Calls

OSDA — API

SQL

DAE Jobs, callbacks

DAE
(Data Acqusition Engine)
SQL & scaling

Database (Sybase)

Tim Bolshakov
2/3/2004

Plots From Supertable – Elements of Luminosity

Monitoring SDA itself

- **Complex system – lots of places for things to wrong**
  - Front ends – instrumentation, MADCs, break
  - DAQ process itself
    - Node down
    - Hung process
    - Sequencer mistakes
    - etc.
  - Problems in the analysis code
- **Needs constant monitoring**
  - Error logs from SDA OACs
  - Store checker
  - User complaints

Store Checker

- Purpose
  - Monitor instrumentation and DAQ
  - Monitor accelerator performance
- Checks SDA data for specified cases/sets
  - Min < device value < max
  - Min < (difference in time of 2 devices) < max
  - Min < (difference in value of 2 devices) < max
- Jobs run automatically every store
- "Standard" and "private" lists
- Results on WWW for "standard" list
- Lists on WWW

Error logs from SDA data acquisition very important

Java is a Good Thing

- Easy to learn – and all the school kids learn it. No memory management headaches.

- Java Web Start programs make maintaining and deploying code much easier.

- Platform independent, so moving to new faster machines is easy.

- Easy WWW integration – really important for access given firewalls.

- Lots of tools for human interfaces

- Division into client side and server side code makes it easy to spread the load, but you can lose this advantage in communications overhead if not careful.

Java is a Bad Thing

- Speed – particularly if you don't think about it in advance.

- Poor tools for interactive scientific graphing and fitting.

- Packages like Origin7 and Mathematica don't have suitable Java interfaces.

- There is a learning curve, even if it is relatively short.

Access to Front End Data

- All of the intelligent front ends like SBD, flying wires, FBI, synclite, need access to the raw data for detailed study to develop and understand the algorithms. This means ways of getting and storing the raw data. (Eugene Lorman's talk.)

- This is really important!

Lessons

- SDA is a success, but the system is very complicated. It is very time consuming to keep on top of things.

- Java is a good thing.

- Our understanding of needs change – need flexibility and extensibility .

- More data and more compute capability is (almost always) useful in ways we never anticipated.

- Still lacking information in SDA and analysis tools
  - ➢ Tunes, chromaticities
  - ➢ More magnet settings, readbacks
  - ➢ BPM and orbit analysis tools